APPENDIX A

SOFTWARE APPENDIX

```
# fullcal.awk                                                    #
# takes input from a POLYchip CEL file (115 x 130) and           #
# extracts ratio information for every block on the chip #

BEGIN{
 ratpatcutoff = 1.2
 pattoggle = "yes"
 base[0]="T"
 base[1]="G"
 base[2]="C"
 base[3]="A"
 name[0,0] = "WI-563"
 hex[0,0] = "CTAGCC"
 name[1,0] = "WI-567"
 hex[1,0] = "TCAGAG"
 name[2,0] = "WI-597"
 hex[2,0] = "TGGATA"
 name[3,0] = "WI-681"
 hex[3,0] = "AACTAA"
 name[4,0] = "WI-801"
 hex[4,0] = "CTTGAG"
 name[5,0] = "WI-802"
 hex[5,0] = "CATCCT"
 name[6,0] = "WI-1099"
 hex[6,0] = "CAGATA"
 name[7,0] = "WI-1147"
 hex[7,0] = "ACGAGC"
 name[8,0] = "WI-1325"
 hex[8,0] = "CTCTAC"
 name[9,0] = "WI-1417"
 hex[9,0] = "GTCTTT"
 name[0,1] = "WI-1796"
 hex[0,1] = "AAAGTG"
 name[1,1] = "WI-1825"
 hex[1,1] = "GTCTTC"
 name[2,1] = "WI-1879"
 hex[2,1] = "TACTGT"
 name[3,1] = "WI-1888"
 hex[3,1] = "ATGACA"
 name[4,1] = "WI-1912"
 hex[4,1] = "TTCTTT"
 name[5,1] = "WI-1959"
 hex[5,1] = "TCTCGG"
 name[6,1] = "WI-1741"
 hex[6,1] = "GAAGGC"
 name[7,1] = "WI-1760"
 hex[7,1] = "ACCACA"
 name[8,1] = "WI-1799"
 hex[8,1] = "TCGATA"
 name[9,1] = "WI-1973"
 hex[9,1] = "CAAGAG"
 name[0,2] = "WI-1980"
 hex[0,2] = "AACTGA"
 name[1,2] = "WI-2015"
 hex[1,2] = "GACTGT"
 name[2,2] = "WI-2664"
 hex[2,2] = "GGAGAG"
 name[3,2] = "WI-4013"
 hex[3,2] = "CTAGTG"
 name[4,2] = "WI-7567"
 hex[4,2] = "TAGTGA"
 name[5,2] = "WI-11595"
 hex[5,2] = "TAGAGC"
 name[6,2] = "CH4.16"
 hex[6,2] = "GATAAT"
 name[7,2] = "WI-6704"
 hex[7,2] = "ACTCCA"
 name[8,2] = "WI-6731"
 hex[8,2] = "GGCACA"
 name[9,2] = "WI-6787"
 hex[9,2] = "ACAGTT"
 name[0,3] = "WI-6910"
 hex[0,3] = "TAGTTG"
 name[1,3] = "WI-9518"
 hex[1,3] = "TTGATT"
 name[2,3] = "ADH3"
 hex[2,3] = "ATAGTT"
 name[3,3] = "AGT"
 hex[3,3] = "GACTGG"
 name[4,3] = "ALDOB-1"
 hex[4,3] = "TTCTGG"
 name[5,3] = "ALDOB-2"
 hex[5,3] = "CCAGAT"
 name[6,3] = "APOB"
 hex[6,3] = "ACTCCT"
 name[7,3] = "APOE(152T/C)"
 hex[7,3] = "TGTCGC"
 name[8,3] = "APOE(290T/C)"
```

```
hex(8.3) = 'AGTCGC'
name(9.3) = 'ARSB'
hex(9.3) = 'TCGATG'
name(0.4) = 'AT1a'
hex(0.4) = 'CTTCCC'
name(1.4) = 'AT1b'
hex(1.4) = 'GCACTT'
name(2.4) = 'BCL2'
hex(2.4) = 'ACGAGG'
name(3.4) = 'BRCA1a'
hex(3.4) = 'CATCTG'
name(4.4) = 'BRCA1b'
hex(4.4) = 'AGAGAG'
name(5.4) = 'BRCA1c'
hex(5.4) = 'GAAGAG'
name(6.4) = 'D3S2'
hex(6.4) = 'CCAGGT'
name(7.4) = 'D3S11'
hex(7.4) = 'TCTGRR'
name(8.4) = 'D3S12'
hex(8.4) = 'CCAGGG'
name(9.4) = 'DRD2'
hex(9.4) = 'CACTGC'
name(0.5) = 'FABP2'
hex(0.5) = 'GCGACT'
name(1.5) = 'GCK'
hex(1.5) = 'GAGACA'
name(2.5) = 'HT2'
hex(2.5) = 'CTGTGG'
name(3.5) = 'HT4'
hex(3.5) = 'TGCAAT'
name(4.5) = 'HT5'
hex(4.5) = 'ACTCGA'
name(5.5) = 'IGF2'
hex(5.5) = 'GGGACC'
name(6.5) = 'IGHV4-6'
hex(6.5) = 'TCTCGA'
name(7.5) = 'INS'
hex(7.5) = 'TCTACC'
name(8.5) = 'LDLR'
hex(8.5) = 'GGCTAA'
name(9.5) = 'LF79'
hex(9.5) = 'CCAGGG'
name(0.6) = 'LPL'
hex(0.6) = 'AGCTAG'
name(1.6) = 'NCC'
hex(1.6) = 'GCCTGA'
name(2.6) = 'MTHH'
hex(2.6) = 'CCCTGG'
name(3.6) = 'NRAMP'
hex(3.6) = 'CAGATG'
name(4.6) = 'PAR'
hex(4.6) = 'ACATTG'
name(5.6) = 'Per/RDS'
hex(5.6) = 'GAAGGA'
name(6.6) = 'PPP1R1'
hex(6.6) = 'GACTAA'
name(7.6) = 'RDS'
hex(7.6) = 'AGGACG'
name(8.6) = 'a14544'
hex(8.6) = 'TCTGCT'
name(9.6) = 'S180A'
hex(9.6) = 'GGCATG'
name(0.7) = 'TcR-CA1'
hex(0.7) = 'TGCGGT'
name(1.7) = 'TcR-CB22'
hex(1.7) = 'GGCTGG'
name(2.7) = 'TcR-CB23'
hex(2.7) = 'CTCTAG'
name(3.7) = 'TcR-CB24'
hex(3.7) = 'GTGATG'
name(4.7) = 'TcR-CB25'
hex(4.7) = 'GTAGCC'
name(5.7) = 'TcR-CB27'
hex(5.7) = 'ACCTTA'
name(6.7) = 'VB12a'
hex(6.7) = 'ACAGTG'
name(7.7) = 'VB12b'
hex(7.7) = 'CACTCA'
bkgsum = 0
bkgnum = 0
}
```

```
readthis = 1
if ($1 ~ /[A-Za-z]/ || $2 ~ /[A-Za-z]/) readthis = 0
if (readthis == 1) rawdata[$1,$2] = $3
if ($1>2 && $2>4) if ($1<112 && $2<124) if ($1<90 || $2<109)
        {
         px = int(($1-3)/11)
         py = int(($2-5)/15)
         pxo = (11*px)+3
         pyo = (15*py)+5
         mx = $1-pxo
         by = $2-pyo
         block = 3*(int(by/5))+7
         if (by%5 != 4 && mx != 10)
                {
                 sb = base(by%5)
                 sig(px,py,block,sb,mx) = $3
                }
         if (by%5 == 4 || mx == 10)
                {
                 bkgsum += $3
                 bkgnum++
                }
        }
}
END{
 printf ("background = %5.2f\n", bkgsum/bkgnum)
 printf "MARKER\tBSTBLK\tRATIO\t\tDB\tCHECK\t\tPATRAT\n"
 for (py=0;py<8;py++) for (px=0;px<10;px++) if (py < 7 || px < 8)
     {
      m[0] = substr(hex(px,py),1,1)
      m[1] = substr(hex(px,py),1,1)
      m[2] = substr(hex(px,py),2,1)
      m[3] = substr(hex(px,py),2,1)
      m[4] = substr(hex(px,py),3,2)
      m[5] = substr(hex(px,py),3,2)
      m[6] = substr(hex(px,py),5,1)
      m[7] = substr(hex(px,py),5,1)
      m[8] = substr(hex(px,py),6,1)
      m[9] = substr(hex(px,py),6,1)
      center = substr(hex(px,py),3,1)"/"substr(hex(px,py),4,1)
      pentamer = m[0]""m[2]"["center"]"m[6]""m[8]
      header = "("px+1","py+1")" " name(px,py) "\n" pentamer "\n"
      headprint = 0
        {
         for (j=0;j<=2;j++)
                {
                 block = (3*j)+7
                 num2 = 0
                 den2 = 0
                 num1 = 0
                 den1 = 0
                 x2 = 0
                 n1 = 0
                 n2 = 0
                 for (f=0;f<5;f++)
                        {
                         maxhi(px,py,block,f) = 0
                         for (g=0;g<4;g++) maxlo(px,py,block,g,f) = 0
                        }
                 for (k=0;k<=9;k++) for (b=0;b<=3;b++)
                        {
                         z = int(k/2)
                         signal = sig(px,py,block,base(b),k)
                         omit = 0
                         if (m[k] ~ base(b)) omit = 1
                         if (omit == 1)
                                {
                                 q = maxhi(px,py,block,z)
                                 if (signal > q) maxhi(px,py,block,z)=signal
                                }
                         if (omit == 0)
                                {
                                 q = maxlo(px,py,block,b,z)
                                 if (signal > q) maxlo(px,py,block,b,z)=signal
                                 if (k%2 == 0)
                                        {
                                         num2 += signal
                                         x2 += (signal)^2
                                         n1++
                                        }
                                 if (k%2 == 1)
                                        {
                                         den2 += signal
                                         x2 += (signal)^2
                                         n2++
                                        }
                                }
```

```
                if (cmit == 1) if (kees || kee5)
                        {
                        if (base[b] == substr(nex(px,py),3,1))
                                {
                                numi += signal
                                }
                        if (base[b] == substr(nex(px,py),4,1))
                                {
                                deni += signal
                                }
                        }
                }
        maxnisum = 0
        for (f=0;f<5;f++)
                {
                maxnisum += maxni[px,py,block,f]
                }
        maxniav = maxnisum/5
        maxiosum = 0
        for (g=0;g<5;g++) for (v=0;v<4;v++)
                {
                maxiosum += maxio[px,py,block,v,g]
                }
        maxiosav = maxiosum/16
        maxrat = maxniav/maxiosav
        num = ((numi/2)-(num2/n1))
        if (num < 0) num = 0
        den = ((deni/2)-(den2/n2))
        if (den <= 0) den = 0.001
        ratio = num/den
        max = numi/2
        if (deni/2 > max) max = deni/2
        n = ni+n2
        stdvxnum = ((n*x2)-(numi+den2)^2)
        if (stdvxnum < 0) stdvx = 0
        stdvx = (stdvxnum/(n^2))^(0.5)
        if (maxrat > ratpatcutoff || pattoggle == "no")
                {
                if (headprint == 0)
                        {
                        printf header
                        headprint = 1
                        }
                printf "\t20/"block"\t"
                printf ("%1.2f\t", ratio)
                if (ratio < 10000) printf "\t"
                rat = ratio
                if (ratio == 0) rat = .00001
                lograt = log(rat)/log(10)
                printf ("%2.2f\t", 10*lograt)
                printf ("%2.2f", max/stdvx)
                if (max/stdvx < 2) printf "\tFAIL\t"
                if (max/stdvx >= 2) printf "\t\t"
                printf ("%2.2f", maxrat)
                if (maxrat > ratpatcutoff) printf "\t"GOODFAT""
                printf "\n"
                }
        }
    }
}
```

United States Patent & Trademark Office

Office of Initial Patent Examination -- Scanning Division

Application deficiencies found during scanning:

☐ Page(s)_____ of _____ were not present for scanning. (Document title)

Page Nº 23-26 of specification are appendix.

☐ Page(s)_____ of _____ were not present for scanning. (Document title)

☐ ***Scanned copy is best available.***